VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Informační systém pro hodnocení firem a pro správu domu

Information System for a Company Evaluation and for the House Administration

2013 Radim Holek

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Radim Holek**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Informační systém pro hodnocení firem a pro správu domu
Information System for a Company Evaluation and for the House Administration

Zásady pro vypracování:

Cílem této práce je vytvořit dva vzájemně propojené informační systémy. První systém bude umožňovat jednoduše hodnotit firmy a druhý systém bude sloužit pro správu domů a bude využívat data z prvního systému při sestavování plánu oprav. Data v prvním systému budou uložena jako tzv. Linked data a bude je tedy možné strojově číst a odkazovat se na ně.
Očekávané funkce systému pro správu domu:
1. Vyúčtování spotřeby vody, tepla, elektřiny společných prostor.
2. Řízení přístupu do jednotlivých částí aplikace.
3. Autentifikace s využitím OpenId a OAuth.
4. Možnost rozesílání hromadných emailů uživatelům.
5. Propojení se službou SIPO.
6. Sestavování plánu oprav a návrhy na firmy, které je budou provádět.
7. Možnost předběžně hlasovat u vypsaného výběrového řízení.
8. Vykreslování grafů spotřeby a cen za jednotlivé služby.

Funkce systému pro hodnocení firem: evidence firem a jejich hodnocení.

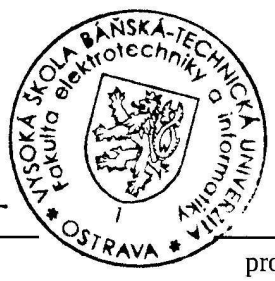
Seznam doporučené odborné literatury:

Podle pokynů vedoucího bakalářské práce.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Radim Bača, Ph.D.**

Datum zadání: 16.11.2012
Datum odevzdání: 07.05.2013



doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 7. května 2013 .....

Abstrakt

Cílem bakalářské práce je návrh a implementace informačního systému pro správu bytových domů. Tento systém má zjednodušit práci při činnostech souvisejících se správou domu. Může se jednat například o roční vyúčtování služeb nebo plánování oprav na domě, proto je součástí této práce také druhý informační systém, který poskytuje údaje o firmách ve formě Linked data. Kromě funkční implementace systémů včetně dokumentace, je součástí bakalářské práce také představení jednotlivých technologií a přístupů použitých při vývoji systémů. Klíčový je koncept Sémantického webu, a jeho implementace s využitím Linked data, a autorizační protokoly pro Internetové služby. Další částí je představení a srovnání technologií ASP.NET.

Klíčová slova: Správa domu, Hodnocení firem, SIPO, Sémantický web, Linked data, ASP.NET, WebForms, MVC, OpenId, OAuth

Abstract

The main goal of this thesis is to design and implement information system for the House Administration. This system is to simplify work in activities related to the administration of the house. This could include annual billing of services or scheduling repairs on the house, therefore the part of this thesis is also dedicated to the second information system, which provides information about companies in the form of Linked Data. In addition to functional systems implementation including documentation, a part of the thesis also shows the various technologies and approaches used to develop systems. The key part is the concept of the Semantic Web, and its implementation using Linked data, and authentication protocols for Internet services. Another part is to introduce and compare technologies of ASP.NET.

Keywords: House Administration, Company Evaluation, SIPO, Semantic web, Linked data, ASP.NET, WebForms, MVC, OpenId, OAuth

Seznam použitých zkratek a symbolů

SIPO	–	Soustředěné inkaso plateb obyvatelstva
HTML	–	HyperText Markup Language
XML	–	Extensible Markup Language
MVC	–	Model-View-Controller
RDF	–	Resource Description Framework
W3C	–	World Wide Web Consortium
URI	–	Uniform Resource Identifier

Obsah

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

1.1 Cíle

Hlavním cílem bakalářské práce je navrhnout a vyvinout informační systém, který má pomoci při správě domu. Do systému bude možné vkládat organizace, vést evidenci obyvatel, provést vyúčtování, atd. Bude také umožňovat přístup obyvatel do systému. Součástí práce je i druhý systém, který bude poskytovat údaje o firmách provádějících opravy na domech ve formě Linked data.

Součástí práce je i prezentace technologií použitých při vývoji systému. Jedná se hlavně o autentifikační protokoly OpenID a OAuth. Dále představení a porovnání serverových technologií .NET frameworku. Informační systém bude umožňovat zadávání předpisů plateb do SIPO, část práce je proto věnována i této problematice. Další částí práce je prezentace konceptu Sémantického webu a Linked data a jeho implementace.

1.2 Důvod vzniku práce a hlavní funkce

Bakalářská práce vzniká z důvodu nedostatku na trhu. Lze sice najít informační systémy, které slouží ke správě domu, ale žádný nebo jen hrstka z nich , umožňuje přístup k informacím v systému také obyvatelům. Tato funkčnost je výhodná zejména proto, že obyvatelé si mohou sami zkontrolovat své údaje. Ověřit si je mohou i nyní, ale musí kontaktovat svého správce, který jim údaje sdělí a nemůžou si je ověřit sami. Je to tedy výhodné pro správce i obyvatele.

Další funkcí je vkládání plánovaných oprav do systému. Ke každé opravě je možné vkládat nabídky firem a zde přichází na řadu již zmiňovaný druhý informační systém, který bude poskytovat údaje o firmách.

1.3 Struktura práce

Počáteční kapitoly jsou věnovány teoretické části, jsou zde představeny použité technologie a případné ukázky. Kapitola 4 je věnována analýze, návrhu a implementaci systému pro hodnocení firem a kapitola 5 systému pro správu domu. Je zde popsána funkční a datová analýza, návrh a implementace.

2 Technologie použité při vývoji systémů

2.1 SIPO

SIPO (Soustředěné inkaso plateb obyvatelstva) je službou České pošty, která spočívá v inkasování plateb od fyzických osob ve prospěch osob právnických, které mají s Českou poštou smlouvu. Důvodem zřízení této služby může být zjednodušení plateb. Pro fyzické osoby je výhodou, možnost zaplatit pouze jednu platbu, která pokryje více služeb. Pro organizace je výhoda, pouze zadat požadavek k inkasování určité částky a vše ostatní přenechat na zprostředkovateli.

Jelikož může často docházet ke změně výše plateb, které právnické osoby inkasují od fyzických osob, je potřeba možnost elektronické komunikace mezi Českou poštou a organizacemi. Pro komunikaci Česká pošta zvolila formu strukturovaných textových souborů v předem známém formátu[1]. Jedná se o dva soubory. V prvním souboru každý řádek reprezentuje jeden záznam, který zadává požadavek na platbu od daného klienta, v daném termínu a v určité výši. V druhém souboru se pak nachází údaje o prvním souboru (počet záznamů, číslo organizace, . . .). Oba soubory, se nakonec přidají do jednoho „zip“ souboru a předají České poště.

Způsob předání souboru je stanoven ve smlouvě mezi organizací a Českou poštou. Může jím být třeba osobní předání na pobočce České pošty, např. disketou nebo jiným přenosným médiem. Nebo předání přes internet, např. emailem na adresu stanovenou ve smlouvě, v tomto případě musí být soubor předán v zašifrované formě, jinak není přijat ke zpracování. Pro zašifrování souboru poskytuje Česká pošta program Crypta, který předá organizaci při uzavření smlouvy.

Po skončení inkasního období, předá Česká pošta organizaci soubor, který obsahuje informace o provedených inkasních platbách. Soubor předává tak, jak stanovuje smlouva (může být i v zašifrované formě). Opět se skládá ze dvou souborů. Kde jeden slouží jako průvodka a druhý obsahuje záznamy pro platby, které byly provedeny.

2.2 Autentifikační protokoly

V dnešní době, kdy vysokým tempem narůstá počet internetových služeb, aplikací a informačních systému a naprostá většina z nich potřebuje z důvodu poskytnutí lepších služeb ověřit identitu uživatele, je pro uživatele náročné pamatovat si přihlašovací údaje ke všem těmto službám. Samozřejmě si je mohou uložit nebo zapsat, ale toto řešení představuje bezpečnostní riziko.

Mnohem elegantnějším řešením je snížení počtu přihlašovacích údajů na jeden či několik málo účtů u ověřených autorit (Google, Microsoft, Facebook, . . .), kterým se uživatelé nebojí svěřit svá data a jednoduše si zapamatují přihlašovací údaje. A pro ověření identity v ostatních systémech využít služeb této autority.

Pro tuto distribuci přihlašování je nutno zavést určitá pravidla, aby bylo možné pomocí jednoho programového kódu, využit služeb více autorit. Pro toto existují dva hlavní protokoly starší OpenID protokol a o něco novější OAuth protokol (oba v současnosti ve verzi 2). Protokoly se od sebe liší, OpenID slouží především k ověření pravosti (au-

tentifikaci) uživatele, naproti tomu OAuth poskytuje autorizaci, tzn. že aplikace na sebe může vzít identitu uživatele [3, 4]. S OAuth je možné například přidávat příspěvky na Facebook, pokud to uživatel povolí. Z tohoto důvodu je pro každého OAuth poskytovatele nutno přepisovat značnou část programového kódu, kdežto u OpenID bude jeden kód fungovat pro všechny poskytovatele.

Pro platformu .NET a programovací jazyk C# existuje knihovna tříd DotNetOpenAuth, která slouží k implementaci obou zmíněných protokolů.

2.2.1 OpenID

Protokol OpenID vznikl v roce 2005 a od roku 2007 je pod správou organizace OpenID Foundation[2].

Uživatel se musí nejprve zaregistrovat u nějakého poskytovatele OpenID, který po registraci poskytne uživateli jedinečný identifikátor (User Identifier), což je typicky URL adresa.

Postup přihlášení je zobrazen na obrázku 1. Pro přihlášení do systému zadá uživatel svůj identifikátor. Aplikace vytvoří požadavek, jehož součástí je identifikátor, seznam požadovaných údajů (Jméno, Příjmení, Email, ...), návratová adresa, atd. A s tímto požadavkem pak uživatele přesměruje k poskytovateli, kde dojde k autentifikaci.

Po ověření na straně poskytovatele, dojde k přesměrování na návratovou adresu zadanou v požadavku (pokud není zadaná, dojde k přesměrování na adresu, odkud přišel požadavek). Návratová zpráva může mít různé stavy:

1. Authenticated: Přihlášení bylo úspěšné a došlo k předání všech požadovaných informací. Je možné tyto údaje načíst a pracovat s nimi.

2. Canceled: Došlo k úspěšnému přihlášení uživatele, ale ten odmítl poskytnout některé nebo všechny dodatečné informace.

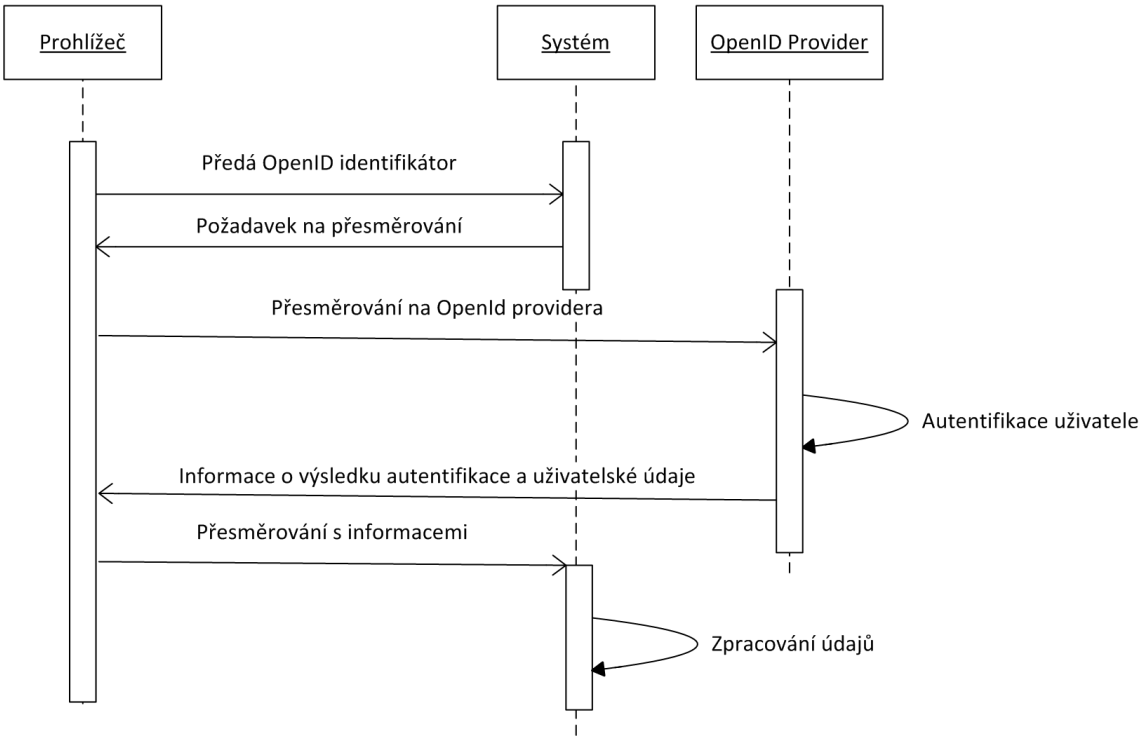
3. Failed: Uživatel nebyl úspěšně ověřen.

2.2.2 OAuth

Jak již bylo zmíněno dříve, OAuth není primárně určen pro autentifikaci, ale k autorizaci. Je možné jej použít k jakési pseudo-autorizaci. Aplikace na sebe jednoduše vezme identitu uživatele a potřebná data si získá sama. Z pohledu uživatele, který požaduje pouze autentifikaci, je tento způsob mnohem méně bezpečný oproti OpenId, kde aplikace získá pouze data, které ji uživatel povolí. Další nevýhodou je minimální interoperabilita mezi poskytovateli a přílišná složitost ve srovnání s OpenID.

První verze pochází z roku 2006. V současnosti je ve verzi 2.0, která je nekompatibilní s předešlými verzemi.

Autentifikovat uživatele je možné na straně klienta (pomocí Javascriptu) nebo na serveru (tzv. server-flow autentification). Další část práve je věnována pouze autentifikaci na straně serveru [5].



Obrázek 1: OpenID

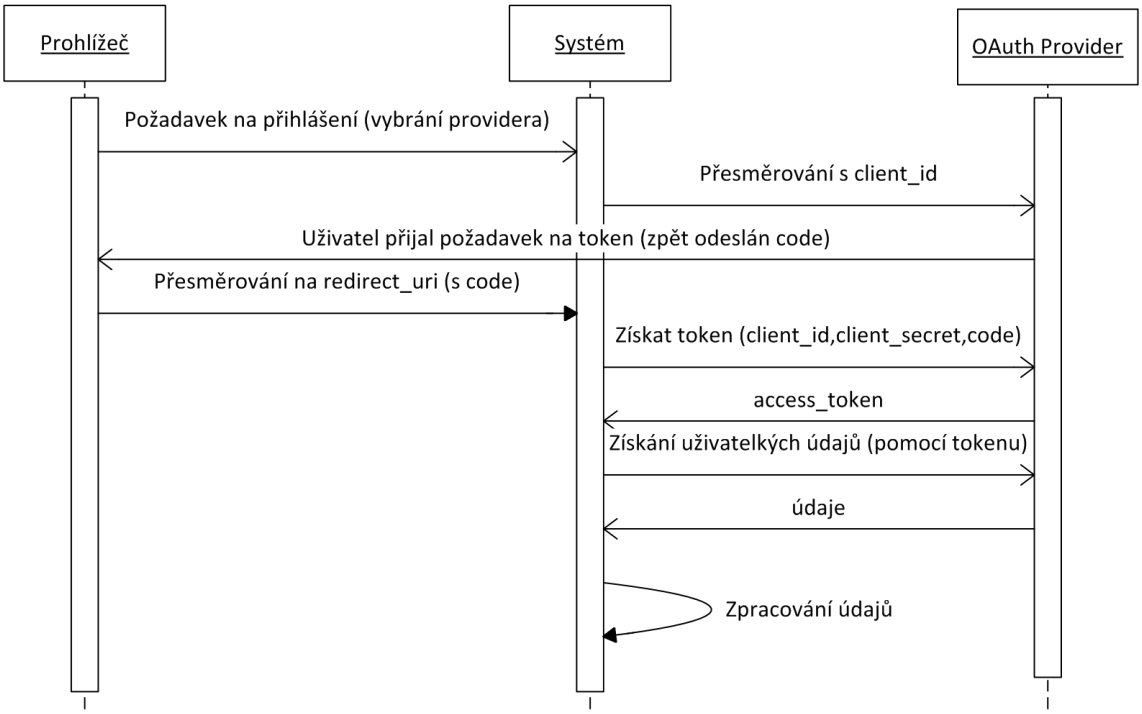
Na rozdíl od OpenID je nutno zaregistrovat aplikaci u poskytovatele, který vygeneruje ClientId (id aplikace) a ClientSecret, které se používají pro ověření při přístupu aplikace. Vetšina poskytovatelů OAuth protokolu má vytvořenou vlastní knihovnu, která obsahuje mimo jiné i klienta pro připojení k aplikaci. Použití tohoto klienta je pak velice jednoduché.

Rozdíl oproti OpenId je vidět na obrázku 2. Uživatel vybere providera a aplikace odešle požadavek na získání access_tokenu. Uživatel toto potvrdí a dotane odpověď s code, který slouží pro ověření aplikace a odešle code do aplikace. Ta si pomocí client_id, client_secret a code požádá o access_token. S tímto tokenem může získat uživatelská data. Rozdíl oproti OpenId je v tom, že nejprve musí získat access_token a až potom může požádat o data. U OpenId požádá o údaje rovnou a to je vše co dostane. Příklad se týká ověření uživatele pomocí Facebooku, ale pro jiné OAuth providery je situace velmi podobná. Vždy musí nejdříve získat access token.

Jelikož je OAuth rozšířený, existuje již mnoho implementací, např. základní šablona pro MVC4 aplikaci ve Visual Studiu 2012, již obsahuje vše potřebné pro autorizaci pomocí několika poskytovatelů (Facebook, Twiter, Microsoft, Google).

2.3 Serverové technologie .NET frameworku - ASP.NET

ASP.NET je součástí .NET Frameworku již od první verze z roku 2002 a je nástupcem Active Server Pages (ASP) [6]. Jako každá jiná technologie v .NET je založen na Common



Obrázek 2: OAuth

Intermediate Language (CIL), což umožňuje programátorům vytvářet aplikace v jakémkoliv jazyce, který je součástí .NET Frameworku.

2.3.1 ASP.NET Web Forms

ASP.NET Web Forms je serverová technologie založená na objektovém přístupu k HTML stránkám. Každá stránka se skládá z tzv. Controls, jedná se o objekty, které reprezentují nějaký prvek zobrazený uživateli. Může to být popisek, textové pole nebo i složitější prvky jako kalendář. Mnoho Controlů je již součástí ASP.NET, ale programátor může vytvářet i vlastní.

Stránka se skládá ze dvou částí. Stránky samotné (soubor s příponou „aspx“) a code behind.

Do stránky se zapisuje HTML kód, který chceme zobrazit, Controly, ať už vlastní nebo předdefinované a také program psaný v jakémkoliv jazyce .NETu (převážně C# a Visual Basic) a uzavřený ve speciálním bloku ohraničeném „<%“ a „%>“. Pro HTML prvky, které je potřeba měnit na straně serveru je nutno přidat atribut „runat=”server”“, prvek pak bude přístupný přes hodnotu atributu ID. Druhou částí stránky je CodeBehind (soubor příponou „cs“), což je třída jazyka C#, která je potomkem třídy Page. V této třídě lze psát programovou část aplikace, reagovat na mnoho různých událostí (PageLoad, ButtonClick, . . .).

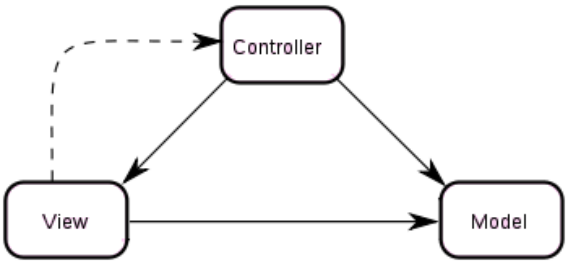
Cílem je, aby šlo s webovou aplikací pracovat podobně jako s aplikací desktopovou (WinForms). Aby toto bylo možné, musí WebForms obsahovat různé techniky, které umožní volání událostí a předávání dat.

Dozvědět se více o této technologii je možné na oficiálních stránkách ASP.NET [7].

2.3.2 ASP.NET MVC

MVC je novou technologií (verze 1.0 pochází z roku 2009, v současnosti verze 4) ASP.NET, která využívá návrhový vzor model-view-controller (obr. 3).

Tento návrhový vzor přepokládá existenci komponent Model (obsahuje data pro zobrazení a doménovou logiku), View (popisuje způsob zobrazení dat a komunikaci s uživatelem) a Controller (zajišťuje komunikaci mezi View a Modelem). Přičemž Model si není vědom existence View ani Controlleru.



Obrázek 3: Návrhový vzor MVC

ASP.NET se využívá tento návrhový vzor rozšířený přes celou webovou aplikaci. Aplikace se skládá ze tří druhů komponent. První jsou Controllery, které obsahují několik akcí (Action) a pro každou z nich může existovat View. Model může existovat pro jednu akci nebo být rozprostřen přes více akcí.

2.3.2.1 Controller Controller je komponenta, která obsahuje jednu třídu (název třídy je název Controlleru s připojeným slovem „Controller“). Tato třída obsahuje několik metod (každá metoda představuje jednu akci), které vykonávají nějakou funkci a vrátí uživateli výsledek. Výsledkem může být HTML dokument (generován pomocí View), soubor, přesměrování na jinou akci, atd.

2.3.2.2 Model Model obsahuje doménovou logiku aplikace, může komunikovat s jinými objekty nebo obsluhovat komunikaci s databází. Jedná se o klasickou třídu, která může obsahovat metody, vlastnosti, atributy, atd.

2.3.2.3 View View využívá technologii Razor. Jedná se o HTML dokument, který může obsahovat některé prvky programovacího jazyka (každý blok kódu je označen @). Lze tedy získat data z metod a vlastností objektů, která se pak zobrazí uživateli. Razor

prostě vloží navrácenou hodnotu metody na místo volání. Je možné používat statické metody, vytvářet vlastní objekty, ale hlavně využívat metody Modelu, který je specifikován na začátku View a předáván z Controlleru.

Více se lze dozvědět na oficiálních stránkách ASP.NET [8].

2.3.3 Porovnání technologií

Porovnání těchto dvou technologií není příliš jednoduché, každá má své výhody i nevýhody. A každá je vhodná pro jiné použití, WebForms se hodí na menší informační systémy a prezentační webové stránky, ale pro velké informační systémy je vhodnější použít MVC.

Důvodů je několik. Tím nejdůležitějším je asi přehlednost kódu, protože rozdělení úlohy do tří částí místo dvou umožní lepší spravování. Je vhodnější také z pohledu web designu, jelikož Razor je v podstatě čisté HTML (a Javascript) je jednodušší se jej naučit (není nutno se učit ASP.NET Controly) a mnohem elegantněji vyřešit např. vypsání všech prvků do tabulky (cyklus „foreach“ přímo v Razoru). Pokud web design dělá někdo jiný, než kdo dělá programovou část, může si sám vybrat, která data budou prezentovány a kde, kdežto u WebForms by musel do stránky vložit prvky, přiřadit jim ID a pak požádat programátora, aby je naplnil z CodeBehind. Nicméně výhodou WebForms je možnost použití designeru ve Visual Studiu, pro MVC žádný designer neexistuje.

MVC poskytují z pohledu uživatele „pěknější“ URL. MVC ji totiž poskytují ve tvaru doména/Controller/Action/parametr, ale WebForms používá URL v „klasickém“ tvaru (cesta k souboru z kořenové složky aplikace s příponou aspx).

WebForms jsou již mnoho let prověřenou technologií, naproti tomu MVC jsou mladé a stále se vyvíjející. Proto lze u WebForms čekat vetší životnost (nebude docházet ke změnám v technologii).

3 Sémantický web

V dnešní době jsme obklopeni ohromným množstvím dat. A to hlavně díky Internetu, kde může kdokoliv vkládat nová data a údaje. A vzhledem k tomuto množství by bylo lepší nechat je zpracovávat stroji. A proto musí být strojově čitelná. Kromě ušetření práce s vyhledávání a analýzou je velkou výhodou strojově čitelných dat jejich jednoduchá znovu-použitelnost.

V současnosti poskytují některé společnosti a státní instituce svá data pro strojové zpracování a to ve formě různých API (Application Programming Interface) a Webových služeb. Důležité je slovo „různých“, protože každá tato služba má vlastní formu a neexistuje standartizace, což je nepraktické.

Například se může jednat o údaje ekonomiky České republiky. V tomto případě by bylo vhodné uvést kromě ekonomických informací o ČR i další základní údaje. A proto je vhodné použít data, která jsou již dostupná, než sepisovat vlastní.

Samozřejmě není problém přidat k datům odkaz na údaje o České republice, ale v tomto případě si musí případný zájemce data, která můžou být dokonce i v jiném formátu, sám zobrazit. Toto není příliš vhodné, protože zájemce pak musí data získávat z různých zdrojů nebo z jiných formátů. Ideálním případem by bylo, kdyby se zájemci zobrazily údaje o ekonomice ČR a ostaní informace najednou. Pokud by nebylo možné zobrazit je najednou (z důvodu objemnosti dat), tak by se alespoň zobrazily ve stejném fomátu.

Tuto ideu popisuje pojem „Sémantický web“, který je považován za součást Webu 3.0 [9, 12, 14]. Hlavní myšlenkou je vytvořit web, kde budou počítače schopny analyzovat veškeré informace a vztahy mezi nimi a poskytovat je uživateli.

V současnosti, kdy web tvoří z největší části HTML dokumenty, je strojově možné pouze hledat klíčová slova. Samozřejmě je možné vytvořit stroj, který se pokusí pochopit sémantiku webu a získat informace, ale toto je složité a ne příliš běžné. Pokud budou data strojově čitelná, je vytvoření tohoto stroje mnohem jednoduší a efektivnější. Pro představu by bylo možné vytvořit vyhledávač, který by místo odkazů na stánky s odpovědí, poskytl odpověď přímo. O takové způsoby vyhledávání se snaží (někdy i docela úspěšně) vyhledávače již dnes, toto by jim práci však značně ulehčilo.

Nicméně Sémantický web je pouze myšlenkou, vizí, jak data publikovat. A Linked data jsou konkrétní protředky, jak dosáhnout této myšlenky [9, 11]. Lze je chápat jako kolekci propojených datových zdrojů ve standartním formátu.

Typickým přikladem Linked data může být DBPedia (dbpedia.org), která v podstatě zpřístupňuje obsah Wikipedie ve standartizovaném formátu (RDF - viz. 3.1.1). Ale to není vše, navíc propojuje data z Wikipedie s jinými datovými zdroji, např. Geonames (služba, která poskytuje geografické informace), přidáním potřebných odkazů.

3.1 Triplestore databáze (Triplestore)

Předpokladem pro strojovou čitelnost jsou strukturované dokumenty. Sémantický web využívá ontologického (výslovný a formalizovaný popis) datového modelu, který narozdíl od relačního či hierarchického modelu umožňuje také využití sémantiky. Tento

model se využívá v triplestore databázích, což je v podstatě databáze skládající se z RDF dokumentů.

3.1.1 RDF

RDF je skupina specifikací vytvořených organizací W3C (World Wide Web Consortium). Původně byl navržen jako model metadat. Jedná se o soubor trojic předmět-predikát-objekt. Využívá se buď, tzv. N3 (Notation3) formátu (ukázka ve výpise 1), nebo formátu RDF/XML, který modeluje trojice a využitím XML (ukázka ve výpise 2)[10]. RDF/XML je standartní XML dokument, který využívá W3C jmenný prostor RDF (http://www.w3.org/1999/02/22-rdf-syntax-ns#).

Obě ukázky obsahují stejné trojice. Předmětem je Empire Burlesque (reprezentovaný jedinečným jménem „http://www.recshop.fake/cd/Empire_Burlesque“), ke kterému jsou pomocí vztahu (predikátu) přiřazeny objekty. Například se jedná o trojici (Empire Burlesque-cd:artist-Tony Benn) říkající, že pro skladbu Empire Burlesque je umělcem Tony Benn. Je zde celkem pět trojic, ale všechny mají stejný předmět (Empire Burlesque).

```
@prefix cd: <http://www.recshop.fake/cd#>.

<http://www.recshop.fake/cd/Empire_Burlesque>
  cd: artist  "Tony_Benn";
  cd:country "USA";
  cd:company "Columbia";
  cd:price  "10.90"
  cd:year "1985"
```

Výpis 1: Ukázka RDF/N3 dokumentu

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.recshop.fake/cd#">

<rdf:Description
rdf:about="http://www.recshop.fake/cd/Empire_Burlesque">
  <cd:artist>Bob Dylan</cd:artist>
  <cd:country>USA</cd:country>
  <cd:company>Columbia</cd:company>
  <cd:price>10.90</cd:price>
  <cd:year>1985</cd:year>
</rdf:Description>

</rdf:RDF>
```

Výpis 2: Ukázka RDF/XML dokumentu

3.1.2 RDFS a OWL

Samotný formát RDF neposkytuje techniky důležité k semantickému popisu dokumentu. Pro tento popis se využívají další dva jmené prostory (specifikované W3C):

- RDFS (RDF Schema - http://www.w3.org/2000/01/rdf-schema#)
- OWL (Web Ontology Language - http://www.w3.org/2002/07/owl#)

Ontologický datový model využívá pro sémantický popis několik typů prvků:

- Hlavička (Header) - popisuje samotnou ontologii. Není povinná. Příklad ve výpise 3 ukazuje definici ontologie http://www.example.com/auta s názvem a popisem.
- Třída (Class) - reprezentuje skupinu jedinců, kteří mají podoné charakteristiky. Popisuje sémantiku pro strojové čtení. Výpis 4 ukazuje příklad popisující definici třídy http://www.example.com/auta#auto, která je třídou všech automobilů. Ve výpise 5 je ukázána definice třídy http://www.example.com/auta#osobniauto, která je podtřídou třídy http://www.example.com/auta#auto.
- Jedinec (Individual) - reprezentuje jednu entitu, abstraktní nebo konkrétní. Jelikož patří k některé třídě, tak stroj který čte tohoto jedince, ví jak má zacházet s daty. Příklad ve výpise 6 definuje jedince http://www.example.com/auta#fabia typu http://www.example.com/auta#osobniauto
- Vlastnost (Property) - popisují vztahy mezi jedinci. Existují dva typy. Datový a objektový. Datový popisuje vztah mezi jedincem a hodnotou. Objektový popisuje vztah mezi dvěmi jedinci. Výpis 7 ukazuje oba typy vlastností. Datovou ve formě „výrobce“ a objektovou ve formě „stejná barva“.

```
<owl:Ontology rdf:about="http://www.example.com/auta">
        < dc:title >Priklad ontoligie  aut</ dc:title >
        <dc:description>Priklad ontologie popisujici  auta</dc:description>
</owl:Ontology>
```

Výpis 3: Ukázka OWL a RDFS - Hlavička

```
<owl:Class rdf:about="http://www.example.com/auta#auto">
   <rdfs:label>Automobil typ</rdfs:label>
   <rdfs:comment>Trida vsech automobilu.</rdfs:comment>
</owl:Class>
```

Výpis 4: Ukázka OWL a RDFS - Definice třídy

```
<owl:Class rdf:about="http://www.example.com/auta#osobniauto">
   <rdfs:subClassOf rdf:resource="http://www.example.com/auta#auto"/>
   <rdfs:label>Osobni auto</rdfs:label>
   <rdfs:comment>Vsechny osobni automobily</rdfs:comment>
</owl:Class>
```

Výpis 5: Ukázka OWL a RDFS - Definice podtřídy

```
<rdf:Description rdf:about="http://www.example.com/auto#fabia">
   <rdf:type rdf:resource="http://www.example.com/auta#osobniauto"/>
</rdf:Description>
```

Výpis 6: Ukázka OWL a RDFS - Definice jedince

```
<owl:DatatypeProperty rdf:about="http://www.example.com/auta#vyrobce"/>
<owl:ObjectProperty rdf:about="http://www.example.com/auta#stejnabarva"/>

<rdf:Description rdf:about="ttp://www.example.com/auto#fabia">
   <rdf:type rdf:resource="http://www.example.com/auta#osobniauto"/>
   <!-- Datova vlastnost -->
   <auta:vyrobce>Skoda</auta:vyrobce>
   <!-- Objektova vlastnost -->
   <auta:stejnabarva rdf:resource="http://www.example.com/auta#golf"/>
</rdf:Description>

<rdf:Description rdf:about="http://www.example.com/auto#golf">
   <rdf:type rdf:resource="http://www.example.com/auta#osobniauto"/>
   <auta:vyrobce>Volkswagen</auta:vyrobce>
   <auta:stejnabarva rdf:resource="http://www.example.com/auta#fabia"/>
</rdf:Description>
```

Výpis 7: Ukázka OWL a RDFS - Definice vlastností

3.1.3 SPARQL

SPARQL (SPARQL Protocol and RDF Query Language) je dotazovací jazyk nad RDF databází[13]. Není nepodobný jazyku SQL pro relační databáze. Vzhledem k tomu že je relativně mladý (v roce 2008 byla verze 1.0 zařazena do W3C Recommendation), nelze od tohoto jazyka očekávat veškeré funkční možnosti SQL a ani takový výkon. Nicméně se jedná o novou a perspektivní technologii, která má řadu výhod. Je více flexibilní, standatrizovaný a stále vyvíjený.

Ve výpise 8 můžeme vidět dotaz, který vybere všechny trojice obsahující predikát rdf:type a objekt company:companytype a uloží jejich předmět do proměnné ?a. Potom pro všechny hodnoty proměnné ?a vybere trojice, která obsahují předmět ?a a predikát company:ic a objekt uloží do proměnné ?ic. Sekce SELECT potom vrátí všechny hodnoty proměnné ?ic.

Dotaz může začínat klíčovým slovem PREFIX, kde je možné přiřadit jmenný prostor (namespace) k prefixu, je ale možné uvést namespace přímo v dotazu. Dále následuje SELECT, který vybírá proměné pro zobrazení. Potom může následovat FROM sekce, kde se definují zdroje, nad kterými se provádí dotaz, ale enginy vyhodnocující dotazy umožňují uvést zdroje i mimo dotaz. Dále násteduje sekce WHERE, kde se definují podmínky. Podmíky se zadávají v podobě trojic, ve kterých je možno použít proměné. Každá tato proměná je načtená a je možné ji vybrat v SELECT sekci. Součástí WHERE může být i FILTER, který umožnuje zadávat podmínky pro data, která mají být načtena. Nakonec může být uvedena sekce ORDER BY.

SPARQL nepodporuje aktualizace dat, což je logické protože společnosti poskytující data, nebudou ve vetšině případů chtít, aby jim kdokoliv mohl údaje měnit. Pro aktualizace existuje jazyk SPARQL/Update, který vychází z jazyka SPARQL.

```
PREFIX company: <http://www.example.com/company#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

```
SELECT ?ic
WHERE
{
?a rdf:type company:companytype.
?a company:ic ?ic.
}
```

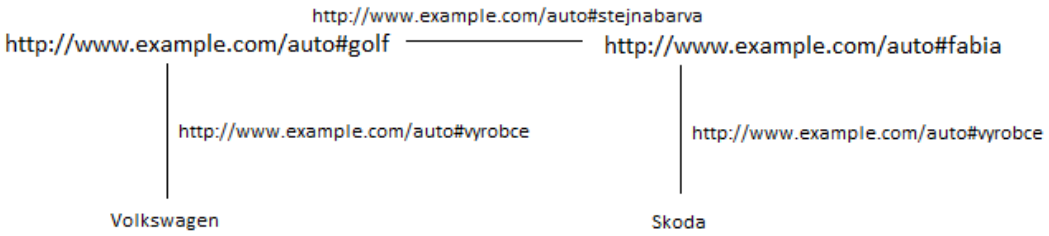
Výpis 8: Ukázka jednoduchého SPARQL dotazu

3.2 Linked data

Technikou, jak implementovat sémantický web (Web of data), jsou Linked data. Jedná se o množství vzájemně propojených triplestore databází.

U zrodu v roce 2007 stál projekt DBPedia.org a několik dalších datových zdrojů. Postupem času se však zapojovali další a další databáze a nyní již obsahuje několik desítek až stovek vzájemně propojených databází. Nemusí se však jednat pouze o triplestore databáze, je nutné pouze použití RDF formátu.

Jak již bylo zmíněno, každá entita potřebuje jedinečné jméno, které bude identifikovat vlastnoti věcí popsaných v dokumentu a jejich vztahy. Protože Linked data jsou určená pro Webovou architekturu, je tímto identifikátorem jedinečná URI (Uniform Resource Identifier). Příklad URI včetně vztahů, které jsou také pojmenovány pomocí jedinečné URI, je na obr. 4.



Obrázek 4: Příklad URI a vztahů mezi nimi.

Takovýto přístup vytvoří a zvýší dohledatelnost dat a tím také jejich znovupoužitelnost.

4 Systém pro hodnocení firem

Systém bude sloužit jako katalog firem, ve kterém mohou zákazníci vyjádřit svou spokojenost či nespokojenost s vykonanou prací.

Hlavním důvodem pro vznik systému v této bakálářské práci je podpora a poskytování dat informačnímu systému pro správu domu. Dále je v rámci implementace tohoto systému představena technologie ASP.NET WebForms a koncept Linked data.

4.1 Analýza

4.1.1 Funkční analýza

Hlavní funkcí tohoto systému je vedení katalogu firem a hodnocení těchto firem uživateli. Uživatelé tedy mohou nejen prohlížet údaje a hodnocení firem, ale také vytvářet vlastní hodnocení a recenze k firmám a vkládat firmy do systému. Na obrázku 5 je vidět nejdůležitější funkce systému.

4.1.1.1 Uživatelská oprávnění K systému budou moci přistupovat celkem čtyři typy uživatelů. První je nepřihlášený uživatel, který může prohlížet katalog a číst recenze. Přihlášený uživatel, který se přihlašuje buď pomocí účtu na Facebooku, nebo pomocí účtu na Googlu, může navíc hodnotit firmy a psát recenze. Navíc může založit novou firmu, pokud v systému ještě není. Dalším typem je majitel firmy, který může upravovat informace o své firmě. O toho oprávnění je možné požádat administrátora. Poslední typem účtu je administrátor, který může upravovat záznamy o všech firmách a udělovat práva majitele firmy.

4.1.2 Datová analýza

Z předpokládáných funkcí systému vyplývá, že bude potřeba ukládat údaje o firmách a jejich hodnocení a zároveň i informace o uživatelích a oprávněních majitele. A protože budou firmy rozděleny do kategorií, je potřeba ukládat i jejich seznam. Diagram na obrázku 6 obsahuje všech pět entit potřebných pro ukládání těchto dat a jejich vztahy.

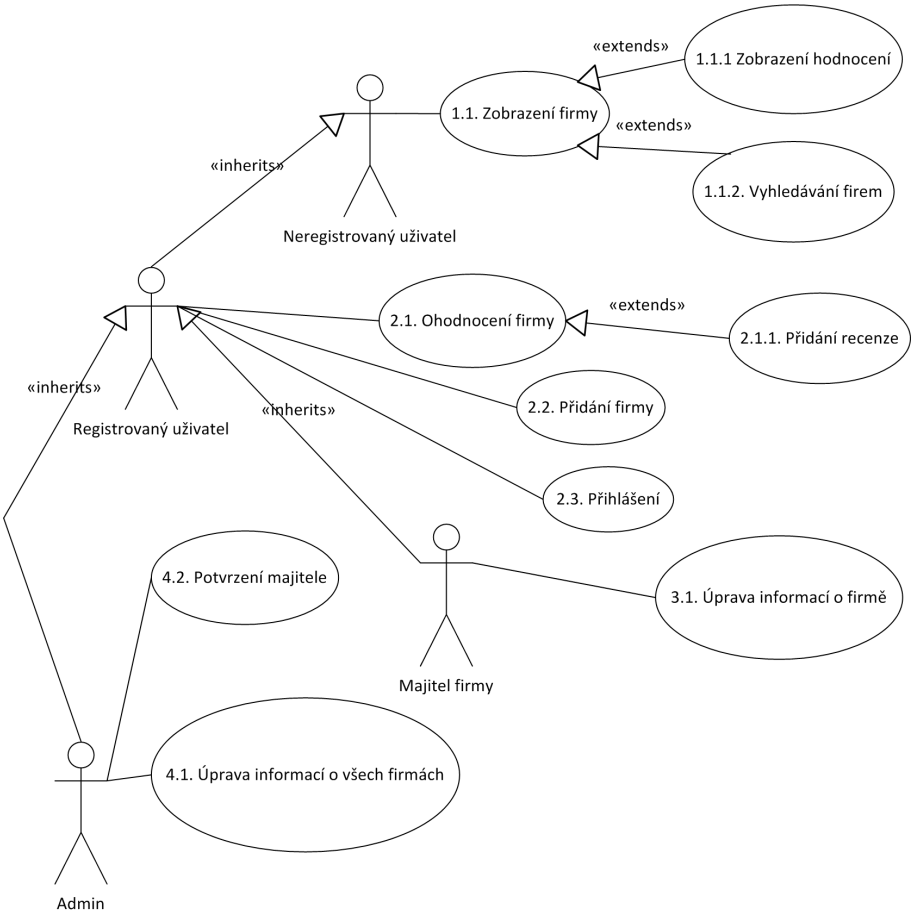
4.1.3 Propojení systému s Obchodním rejsříkem

Původním záměrem bylo propojit tento systém s Obchodním rejstříkem ČR, ale přestože vláda České republiky slíbila zpřístupnění dat Obchodního rejstříku pro strojové zpracování, v době tvorby této práce (jaro 2013) nejsou data stále k dispozici.

4.2 Návrh

4.2.1 Návrh datové vrstvy

Pro ukládání údajů o uživatelích a oprávnění majitelů bude použita relační databáze. Model databáze je na obrázku 7.



Obrázek 5: Funkce systému pro hodnocení firem

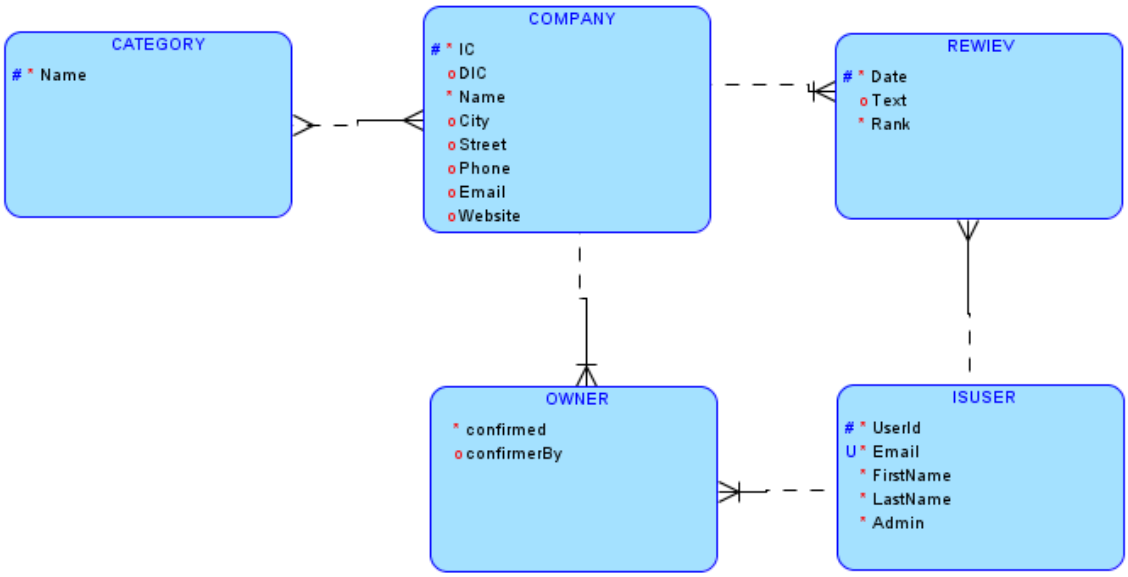
Ve výpise 9 je zapsána ontologie, ve které budou ukládány údaje o firmách. Ontologie obsahuje jednu třídu všech firem companytype, která může mít datové vlastnosti (ic, dic, name, city, street, phone, email, website a dvě datové kolekce categories a reviews). Data budou ukládány do jediného dokumentu, protože to zjednoduší přístup, organizaci dat a dotazování. Nevýhodou tohoto způsobu je možná velká velikost dokumentu, ale protože jedna firma (včetně recenzí) potřebuje asi 10 kB prostoru a předpokládaný počet firem je maximálně v řádu jednotek tisíců, tak celková velikost souboru by neměla překročit 50 MB, což je přijatelné.

Seznam kategorií bude ukládán v jednoduchém XML dokumentu a root elementem categories a podelementy category s názvem kategorie.

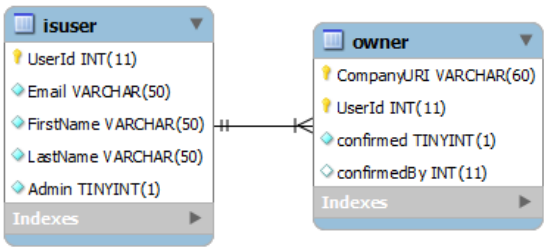
```
<owl:Ontology rdf:about="http://localhost:1496/company#">
  <dc:title>Ontologie hodnoceni firem</dc:title>
  <dc:description>Ontologie hodnoceni firem</dc:description>
</owl:Ontology>
<owl:Class rdf:about="http://localhost:1496/company#companytype">
  <rdfs:label>Typ firma</rdfs:label>
```

```
    <rdfs:comment>Trida firem.</rdfs:comment>
  </owl:Class>
  <owl:DatatypeProperty rdf:about="http://localhost:1496/company#ic" />
  <owl:DatatypeProperty rdf:about="http://localhost:1496/company#dic" />
  <owl:DatatypeProperty rdf:about="http://localhost:1496/company#name" />
  <owl:DatatypeProperty rdf:about="http://localhost:1496/company#city" />
  <owl:DatatypeProperty rdf:about="http://localhost:1496/company#street" />
  <owl:DatatypeProperty rdf:about="http://localhost:1496/company#phone" />
  <owl:DatatypeProperty rdf:about="http://localhost:1496/company#email" />
  <owl:DatatypeProperty rdf:about="http://localhost:1496/company#website" />
  <owl:DatatypeProperty rdf:about="http://localhost:1496/company#categories" />
  <owl:DatatypeProperty rdf:about="http://localhost:1496/company#reviews" />
```

Výpis 9: Ontologie dat systému pro hodnocení firem



Obrázek 6: ER diagram popisující všechny entity v systému pro hodnocení firem



Obrázek 7: Model relační databáze systému pro hodnocení firem

4.2.2 Návrh komponent

Systém bude složen z celkem dvou komponent - Database a UserInterface.

Database je komponenta, která mapuje relační databázi a RDF dokumenty na objekty, provádí také ukládání načtených dat do paměti pro pozdější rychlejší přístup. Zajišťuje veškerou doménovou logiku aplikace. Důvodem vzniku je logické oddělení domény od uživatelského rozhraní a možné znovu využití při vývoji jiného klienta (např. mobilní aplikace) nebo při změně serverové techologie.

UserInterface je komponenta, která využívá komponentu Database. Zprostředkovává komunikaci s uživatelem. Nevykonává žádnou doménovou logiku, kromě obsluhy přihlašování. Obstarává požadavky a odpovědi protokolů OpenID a OAuth, na Database potom odesílá pouze data získána z odpovědi, nikoliv celou odpověď.

4.3 Implementace

4.3.1 Implementace datové vrstvy

Jako relační databáze je použita databáze MySQL 5.5 se storage enginem InnoDB.

4.3.2 Implementace komonent

Pro implementaci je použit .NET Framework 4.5 a programovací jazyk C#. Byly použity i tyto externí knihovny:

- DotNetRDF - knihovna pro práci s RDF dokumenty a vykonávání SPARQL dotazů - http://www.dotnetrdf.org/
- DotNetOAuth - knihovna pro práci s autentifikačními protokoly OpenId a OAuth - http://dotnetopenauth.net/
- MySQL Connector - knihovna pro objektově-relační mapování MySQL databáze - dev.mysql.com/downloads/connector/net/

Komponenta Database je implementována jako knihovna tříd (Database.dll). Je vytvořeno vlastní mapování databáze. Pro ukládání dat do RDF dokumentu je využito DOMu a jazyka XPath. Pro načítaní dat je využito jazyka SPARQL. Komponenta UserInterface je WebForms aplikací.

4.3.3 Požadavky

Z použitých technologií vyplývájí následující požadavky na systém:

- Server : IIS 7.0
- .NET Framework 4.5
- Plná funkčnost systému je zajištěna pouze v prohlížeči Google Chrome verze 25 a vyšší.

5 Systém pro správu domu

Tento systém slouží ke zjednodušení správy domu. Správa domu je služba poskytována vlastníkům jednotek v jednom bytovém domě. Tato činnost zahrnuje převážně roční vyúčtování služeb, vedení účetnictví, zajišťování firem provádějící opravu nebo údržbu na domě a spoustu dalších drobných činností.

Jelikož na trhu existuje již velká řada aplikací, které umožňují vést účetnictví, tento systému tuto funkcionalitu neobsahuje.

5.1 Analýza

5.1.1 Funkční analýza

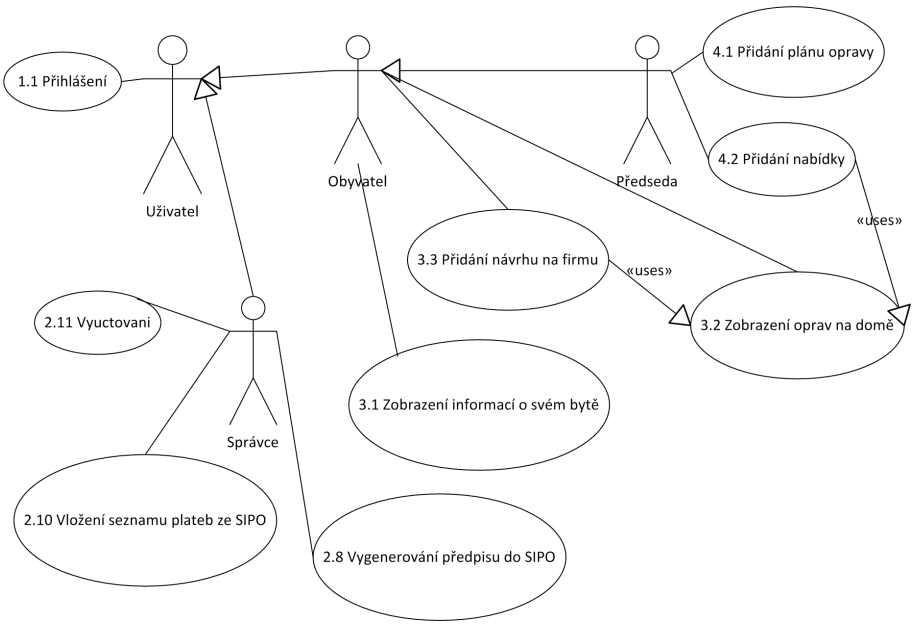
Na obrázku 8 je vidět nejdůležitější netriviální funkce systému. Všechny netriviální funkce lze nalézt na obrázku 1 v příloze. Nejdůležitější funkce jsou:

- 2.8 Vygenerování předpisu do SIPO - správce zadá organizaci a datum, od kterého chce zaznamenat změny a systém vygeneruje zip soubor, který je potřeba předat České poště.
- 2.10 Vložení seznamu plateb ze SIPO - správce nahraje soubor zaplacených plateb, který obdržel od České pošty a systém zapíše zaplacené platby.
- 2.11 Vyučtování - správce může provést vyúčtování služeb. Vybere organizaci, období a seznam služeb, které chce vyúčtovat a systém provede potřebné výpočty. Správci pak vrátí PDF dokument, který obsahuje rozpis vyúčtování pro každý byt zvlášť. Způsob vyúčtování je podrobněji popsán v kapitole 5.1.2.

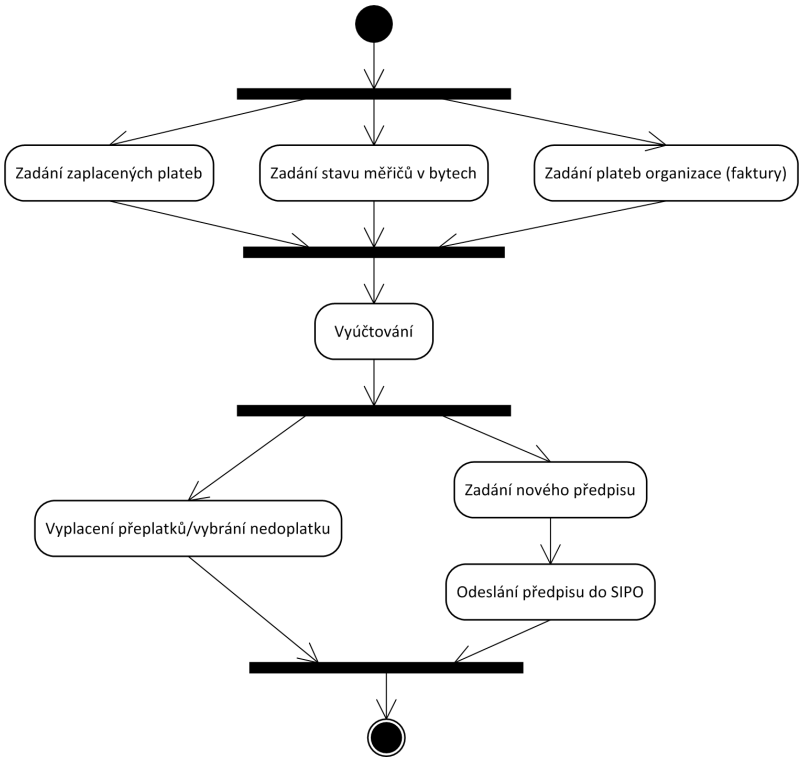
Na obrázku 9 je diagram popisující postup v jednom cyklu vyúčtování, což je vždy jeden rok od 1. ledna do 31. prosince. Během tohoto roku se sbírají informace o zaplacených platbách a fakturách účtovaných organizaci. Po skončení roku se během tří měsíců doplní další faktury, které ještě předtím nebyly k dispozici a stavy měřičů v bytech. Poté dojde k vyúčtování, které je podrobněji popsáno v kapitole 5.1.2. Po vyúčtování, které musí být hotovo nejpozději do konce dubna roku následujícím po vyúčtovaném období, se provede vyplacení přeplatků a vybrání nedoplatků na zálohách a podle výše spotřeby jsou upraveny zálohy a zadány do předpisu k platbě. Pro obyvatele, kteří využívají SIPO je tento předpis nahrán do systému České pošty. Ke změně výše přepisu, zadání stavu měřičů i dalších informací může samozřejmě docházet i během roku, pokud se například změní majitel bytu.

5.1.2 Vyúčtování

Tato kapitola popisuje způsob, jakým se provádí vyúčtování jednotlivých služeb podle platných zákonů České republiky. Běžně se vyúčtovávají služby jako je teplo, což je množství spotřebovaného tepla pro vytápění, které fakturuje teplárenská společnost organizaci, studená voda, teplá voda, která se skládá ze dvou složek vody a tepla pro ohřev, které je



Obrázek 8: Nejdůležitější funkce systému pro správu domu



Obrázek 9: Diagram popisující jeden cyklus vyúčtování

vetšinou součástí faktury za teplo pro vytápění a výsledná částka se počítá pomocí poměrových měřičů, osvětlení společných prostor, výtah, úklid, správa, společná televizní anténa (STA) a další služby podle potřeb organizace.

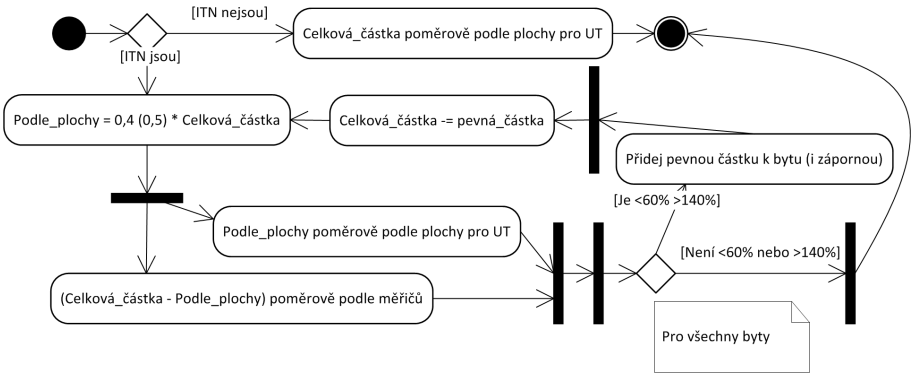
Aby bylo možné správně provést vyúčtování je potřeba pro každý byt zvlášť evidovat tyto údaje:

- Plocha - celková plocha bytu.
- Plocha pro UT - plocha bytu, která se využívá při vyúčtování tepla. Počítá se z celkové plochy podle koeficientu a pro každou místnost zvlášť. Koeficient závisí na počtu otápěných stěn, typu místnosti, atd.
- Plocha pro TUV - plocha bytu, která se využívá při vyúčtování ohřevu vody. Do této plochy se nazapočítává např. balkón, sklepní kóje, atd. .
- Počet obyvatel - počet obyvatel v jednotce (pro každý den zvlášť).
- Spotřeba vody - spotřebu vody bytové jednotky za období (teplá a studená voda zvlášť).
- Zaplacené zálohy - výše plateb zaplacených jako zálohy na služby.

Dále je potřeba mít k dispozici výše plateb organizace za jednotlivé služby a u některých služeb i odebrané množství.

5.1.2.1 Teplo Vyúčtování tepla lze provádět dvěma způsoby v závislosti na tom, zda jsou v domě (v každé místnosti) nainstalovány ITN (indikátory topných nákladů). Pokud jsou nainstalovány, pak se čtyřicet (nebo padesát, podle podle rozhodnutí organizace) procent celkového nákladu účtuje poměrově podle plochy pro UT a zbylých šedesát (nebo padesát) procent celkové částky se účtuje poměrově podle stavu měřičů (které se ještě předtím násobí koeficientem podle umístění místnosti). Žádná částka však nesmí být nižší než šedesát procent průměrné výše platby a vyšší než stočtyřicet procent. Pokud tak nastane je částka zvýšena (snížena) o potřebnou sumu a dojde znovu k přepočítání. Pokud ITN nainstalovány nejsou, tak se celá částka účtuje poměrově podle plochy pro UT. Tento postup je zobrazen na obrázku 10.

5.1.2.2 Studená voda Studená voda se účtuje podle spotřeby. Jelikož vodoměry v bytech nemusí být zcela přesné, může docházet k odchylce mezi součtem spotřeby za jednotivé byty a celkovou spotřebou organizace. Proto se vypočte cena za m^3 (celková cena/celková spotřeba organizace) a každému bytu se naúčtuje cena za jeho spotřebu (cena za m^3 * počet m^3 za byt) a pokud několik m^3 chybí (nebo přebývá), účtuje se tato ztráta podle rozhodnutí organizace. Může to být poměrem podle spotřeby nebo podle počtu osob žijících v bytě.



Obrázek 10: Diagram znázorňující vyúčtování tepla

5.1.2.3 Teplá voda Cena za teplou vodu se skládá ze dvou složek. První složkou je voda, která se účtuje stejně jako studená voda, a druhou složkou je ohřev vody. Sedmdesát procent celkové částky za ohřev je účtováno poměrově podle spotřeby bytů. Zbylých třicet procent je účtováno poměrově podle výměry pro TUV.

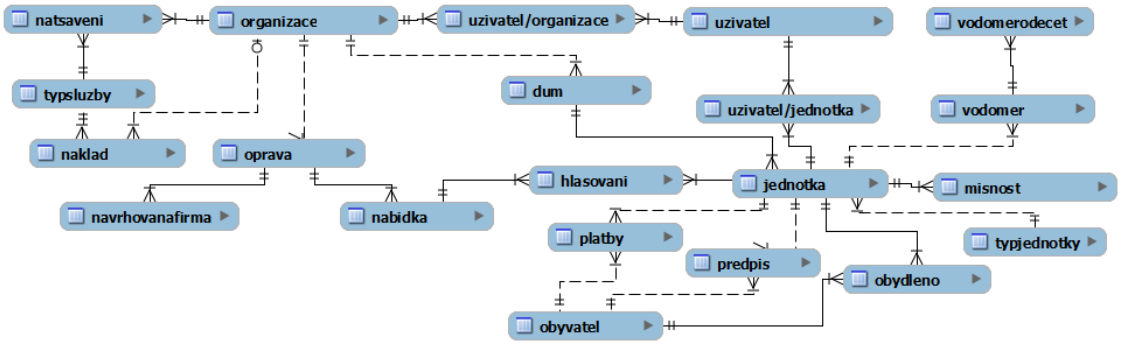
5.1.2.4 Ostatní služby Mezi ostatní služby může patřit osvětlení společných prostor, úklid chodeb, správa, výtah, televizní anténa, atd. Tyto služby se účtují podle stanov organizace a je možné je účtovat podle počtu osob, běžné plochy, nebo za každou jednotku stejně. Je bežné že osvětlení společných prostor, výtah a úklid se účtuje podle osob (pro každý den zvlášť, protože počet osob se může měnit). Správa, televizní anténa za každou jednotku stejně.

5.1.2.5 Zálohy na opravy Jedná o speciální druh služby, který se nezúčtuje při ročním vyúčtování. Slouží jako fond pro budoucí opravy na domě. Každý byt do něj přispívá buď pevnou částkou za byt, nebo podle plochy, anebo kombinací obou (podle stanov organizace). Ačkoliv se tato služba nezúčtuje, je potřeba jej započítat do měsíčního předpisu pro byt.

Po vyúčtování všech služeb, dojde k výpočtu nedoplatku (přeplatku) vzhledem k výši zaplacených záloh.

5.1.3 Datová analýza

Z funkční analýzy vyplývá, že bude potřeba ukládat údaje o celých organizacích i jednotlivých bytech. Všechny potřebné entity jsou zaznamenány na obrázku 11. Jedná se o zjednodušený model, úplný model lze najít v příloze na obrázku 2. Nejdůležitější entitou je Organizace, která reprezentuje organizaci (společenství vlastníků nebo bytové



Obrázek 11: Zjednodušený datový model systému pro správu domu

družstvo). Ke každé organizace existuje entita Nastaveni, která obsahuje nastevení způsobu účtování služeb pro organizaci (každá organizace může mít jiné pravidla) a entita Oprava, která je jednou opravou na domě. Dále jsou k organizaci přidělena přístupová práva pro uživatele (předseda organizace). Druhou důležitou entitou je Jednotka, která reprezentuje jednu jednotku (byt, nebytový prostor, garáž, ...) v domě. S organizací je propojena přes entitu Dum (organizace se může skládat z více domů nebo vchodů). Pro jednotku se evidují místnosti, obyvatelé, předpisy, zaplacené platby a vodoměry, ke kterým se eviduje i jejich stav. Pro jednotku se eviduje ještě hlasování pro nabídku u opravy. K jednotce můžou být přidělená přístupová práva (vlastník jednotky).

5.2 Návrh

5.2.1 Návrh datové vrstvy

Pro ukládání dat bude použita relační databáze vytvořená podle již zmiňovaného ERD (obr. 11).

5.2.2 Návrh komponent

Systém bude rozdělen na dvě komponenty: Domain a UserInterface.

Komponenta Domain bude obsluhovat vetšinu doménové logiky a mapování databáze s využitím existujícího frameworku. Kromě dvou vazebních tabulek bez informace (uzivatel/jednotka a uzivatel/organizace), které budou mapovány na kolekci, bude použito mapování 1:1. Nejdůležitější netriviální třídy budou Vyuctovani, PredpisDoSIPO a tridy s nazvem entityDomain.

Třída Vyuctovani příjmá jako parametr konstruktoru organizaci, období a seznam služeb, které má vyúčtovat, a provede vyúčtování. Metoda GetPDF vrací PDF dokument s vyúčováním. Třída PredpisDoSIPO generuje soubory potřebné pro elektronickou komunikaci se systémem SIPO. Třídy s názvem entitaDomain, kde entita je název entity z databáze, jsou statické a provádějí doménovou logiku (vkládání, mazání a některé další funkce). Ostatní třídy lze nalézt v dokumentaci k systému v elektronické příloze.

UserInterface bude řešit komunikaci systému s uživatelem a bude využívat služeb komponenty Domain.

5.3 Implementace

5.3.1 Implementace datové vrstvy

Jako relační databáze je zvolena databáze MySQL 5.5 se storage enginem InnoDB.

5.3.2 Implementace komponent

Komponenta Domain je knihovna tříd „Domain.dll“ v jazyce C#. Mapuje databázi s použitím ADO.NET entity frameworku.

Komponenta UserInterface je MVC4 aplikací, která obsahuje kromě několika modelů i celkem pět Contorollerů:

- AccountController - obsluhuje přihlašování, registarci a správu uživatelských účtů
- AdministraceController - obsluhuje vytváření uživatelů a přidělování práv, vytváření nových oragnizací, domů, bytů a evidenci obyvatel
- HomeController - obsluhuje společné funkce pro více typů uživatelů, hlavně odesílaní hromadných emailů
- PrehledController - obsluhuje přístup obyvatel, prohlížení údajů a hlasování
- SpravaController - obsluhuje správu domu, přidávání spotřeby, stavů vodoměrů, vyúčtování, tvorbu předpisů a jejich export do SIPO

5.3.3 Použité knihovny

Kromě ASP.NET MVC a dalších technologií .NET Frameworku 4.0 byly při vývoji systému použity i knihovny tříd třetích stran:

- DotNetZip - C# knihovna, která umožňuje jednoduchou práci se zip archívy - http://dotnetzip.codeplex.com/
- iTextSharp - knihovna, která zjednodušuje tvorbu a manipulaci s PDF dokumnety - http://www.sourceforge.net/projects/itextsharp/
- DotNetRDF - vykonávání SPARQL dotazů - http://www.dotnetrdf.org/
- jQuery - JavaScriptová knihovna - http://www.jquery.com/
- fancybox - JavaScriptový plugin, který umožňuje zobrazení HTML obsahu ve stylu „lightbox“ z MacOS - http://www.fancybox.net/
- flotr2 - JavaScriptový plugin, který umožňuje vykreslování grafů s pomocí HTML5 prvku Canvas - http://www.humblesoftware.com/flotr2/

- jquery plugin treeview - vykresluje zobrazení „treeview“ (složky a podsložky) - http://www.bassistance.de/jquery-plugins/jquery-plugin-treeview/

5.3.4 Požadavky

Z použitých technologií vyplývájí následující požadavky na systém:

- Server : IIS 7.0
- .NET Framework 4.0
- Plná funkčnost systému je zajištěna pouze v prohlížeči Google Chrome verze 25 a vyšší.

6 Závěr

Cílem bakalářské práce bylo vyvinout dva propojené informační systémy. Jeden měl poskytovat katalog firem a jejich hodnocení a druhý měl pomoci při správě domu. Měly být na sobě téměř nezávislé, jen systém pro správu domu měl využívat informace o firmách z druhého systému ve formě Linked data, ale funkčnost systému na tom neměla záviset. Součástí práce mělo být i využití technologií OpenID, OAuth a propojení se službou SIPO.

Tyto funkce byly splněny. Byl vyvinut systém pro hodnocení firem, který umožňuje autentifikaci pomocí protokolů OpenID a OAuth. Také poskytuje informace o firmách ve formě Linked data, nicméně zvolené údaje, informace a hodnocení firem, nebyly příliš vhodné pro tuto implementaci a to z důvodu, že Sémantický web a Linked data nejsou příliš vhodné pro často se měnící informace. Proto by bylo vhodnější zvolit jako data pouze informace o firmách a hodnocení do nich nevkládat. Vývoj celého systému byl prováděn na doméně „localhost“, ale protože Linked data používají pro popis entity jedinečné URI, bylo by při reálném provozu potřeba zaregistrovat doménu a URI tvořit nad ní.

Systém pro správu domu byl také implementován. Obsahuje autentifikaci pomocí protokolů OpenID a OAuth. Prozkoumal a implementoval jsem propojení se službou SIPO. Další funkcí mělo být roční vyúčtování služeb, to jsem prozkoumal, zdokumentoval a implementoval. Systém měl umět vést databázi plánovaných oprav na domě, vypisovat výběrová řízení, přidávat nabídky firem s možností pro nabídku hlasovat. Provedl jsem to tak, že je možné vypsat opravu, která je považována za výběrové řízení, a k ní přidávat nabídky, pro které je možné hlasovat.

Po implementaci obou systému jsem provedl testování. Jednalo se ale pouze o krátké testování, takže pokud by měl být systém nasazen do praxe, vyžadoval by ještě mnoho dalších testů.

K systémům jsem vytvořil dokumentaci, návod k instalaci a k systému pro správu domu ještě uživatelský manuál. Vše lze nalézt v elektronické příloze. Oba systémy jsem nainstaloval na virtuální server pro testování, informace o dostupnosti jsou v elektronické příloze.

V této práci jsem také představil nejdůležitější technologie použité při vývoji systému, ukázal jsem způsob implementace autetifikace pomocí protokolů OpenID a OAuth, představil jsem koncept sémantického webu a Linked data, ukázal příklady použití technologií ASP.NET WebForms a MVC a představil jsem způsob strojové komunikace s Českou poštou pro zadávání příkazů do SIPO.

6.1 Další rozšíření

Přesto, že jsou systémy funkční a obsahují všechny požadované funkce, lze stále vymyslet možnosti, jak je rozšířit.

Systém pro hodnocení firem by mohl být doplněn o autentifikaci i vůči jiným OAuth providerům než Facebook. A až bude Obchodní rejstřík poskytovat strojově čitelné údaje, lze doplnit načítání údajů o firmách z něj.

Systém pro správu domu by mohl také obsahovat přihlašování s jinými providery, což není nijak težké, jelikož jsou implementovány metody pro obsluhu dalších OAuth klientů, stačilo by pouze zaregistrovat aplikaci. Dále by do systému mohla být implementována možnost vyúčtování tepla podle ITN. Další možností by bylo evidovat v systému revize a odesílat připomínky emailem, pokud by platnost revize končila. Do systému by mohlo být přidáno také účetnictví, které by mělo, oproti ostatním účetním programům, výhodu propojení s údaji o správě (vyúčtování, platby, ...). Vyjma vyúčtování tepla podle ITN by tato rozšíření téměř vůbec nezasahovala do struktury databáze a tříd, takže by je bylo možné jednoduše připojit.

7 Reference

[1] *Technické podmínky organizace do sipo* [online]. 07.2008 [cit. 10.3.2013] http://www.ceskaposta.cz/assets/sluzby/penezni-sluzby/cr/P2_Technicke-podminky-organizace-do-SIPO-zmeny.pdf

[2] *OpenID Foundation website* [online]. ©2006-2013 [cit. 30.3.2013] http://www.openid.net/

[3] *OpenID - Wikipedia, the free encyclopedia*[online]. [cit. 27.3.2013] http://en.wikipedia.org/wiki/OpenID

[4] *OAuth - Wikipedia, the free encyclopedia* [online]. [cit. 27.3.2013] http://en.wikipedia.org/wiki/OAuth

[5] *Login for Server-side Apps - Vývojáři společnosti Facebook* [online]. ©2013 [cit. 30.3.2013] https://developers.facebook.com/docs/howtos/login/server-side-login/

[6] *ASP.NET - Wikipedia, the free encyclopedia* [online]. [cit. 6.4.2013] http://en.wikipedia.org/wiki/ASP.NET

[7] *Web Forms : The Official Microsoft ASP.NET Site* [online]. ©2013 [cit. 6.4.2013] http://www.asp.net/web-forms

[8] *MVC : The Official Microsoft ASP.NET Site* [online]. ©2013 [cit. 6.4.2013] http://www.asp.net/mvc

[9] Tom Heath and Christian Bizer (2011) *Linked Data: Evolving the Web into a Global Data Space* (1st edition). Synthesis Lectures on the Semantic Web: Theory and Technology, 1:1, 1-136. Morgan & Claypool.

[10] *RDF Example* ©1999-2013 [online] [cit. 24.3.2013] http://www.w3schools.com/rdf/rdf_example.asp

[11] *Introducing Linked Data And The Semantic Web* [online]. ©2009 [cit. 24.3.2013] http://www.linkeddatatools.com/semantic-web-basics

[12] *Semantic Web - Wikipedia, the free encyclopedia* [online]. [cit. 24.3.2013] http://en.wikipedia.org/wiki/Semantic_Web

[13] *SPARQL - Wikipedia, the free encyclopedia* [online]. [cit. 8.4.2013] http://en.wikipedia.org/wiki/SPARQL

[14] *Data - W3C* [online]. ©2013 [cit. 8.4.2013] http://www.w3.org/standards/semanticweb/data

Seznam příloh

Seznam tištěných příloh

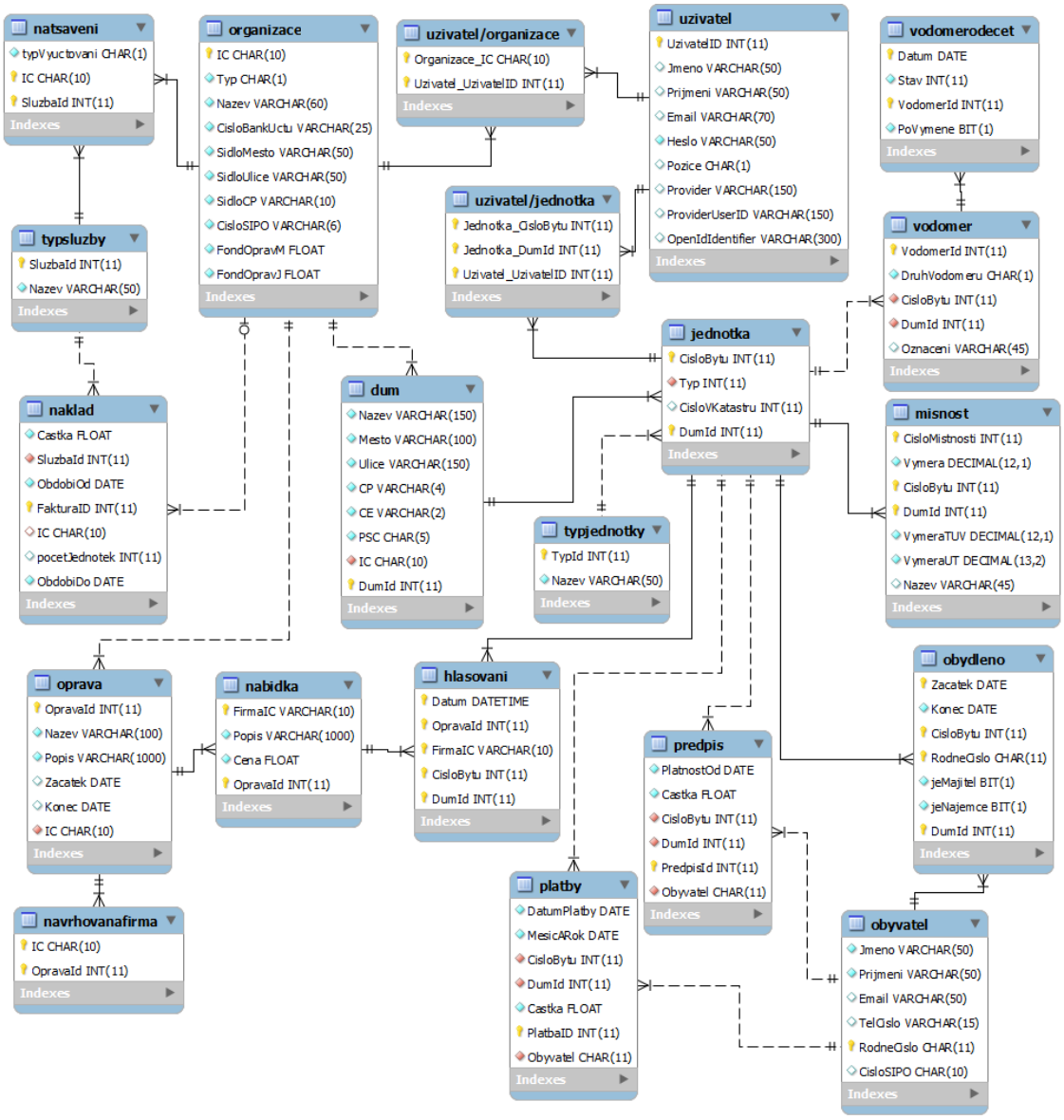
Seznam elektronických příloh

1. Hodnocení firem
   - Instalace - návod a potřebné soubory k instalaci
   - Zdrojové kódy - zdrojové kódy aplikace
   - Dokumentace - dokumentace k aplikaci
2. Správa domu
   - Instalace - návod a potřebné soubory k instalaci
   - Zdrojové kódy - zdrojové kódy aplikace
   - Dokumentace - dokumentace k aplikaci
   - Manuál - uživatelský manuál k systému
3. Systémy k testování - informace pro testování systémů

Funkce systému pro správu domu

Obrázek 1: Všechny netriviální funkce systému pro správu domu

Datový model systému pro správu domu



Obrázek 2: Úplný datový model systému pro správu domu